

Primitive Words and Lyndon Words in Automatic Sequences

Daniel Goč and Jeffrey Shallit

School of Computer Science

University of Waterloo

Waterloo, ON N2L 3G1

Canada

dgoc@cs.uwaterloo.ca, shallit@cs.uwaterloo.ca

July 27, 2012

Abstract

We investigate questions related to the presence of primitive words and Lyndon words in automatic sequences. We show that the Lyndon factorization of a k -automatic sequence is itself k -automatic. We also show that the function counting the number of primitive factors (resp., Lyndon factors) of length n in a k -automatic sequence is k -regular.

1 Introduction

We start with some basic definitions. A nonempty word w is called a *power* if it can be written in the form $w = x^k$, for some integer $k \geq 2$. Otherwise w is called *primitive*. Thus **murmur** is a power, but **murder** is primitive. A word y is a *factor* of a word w if there exist words x, z such that $w = xyz$. If further $x = \epsilon$ (resp., $z = \epsilon$), then y is a *prefix* (resp., *suffix*) of w . A prefix or suffix of a word w is called *proper* if it is unequal to w .

Let Σ be an ordered alphabet. We recall the usual definition of lexicographic order on the words in Σ^* . We write $w < x$ if either

- (a) w is a proper prefix of x ; or
- (b) there exist words y, z, z' and letters $a < b$ such that $w = yaz$ and $x = ybz'$.

For example, using the usual ordering of the alphabet, we have **common** $<$ **con** $<$ **conjugate**. As usual, we write $w \leq x$ if $w < x$ or $w = x$.

A word w is a *conjugate* of a word x if there exist words u, v such that $w = uv$ and $x = vu$. Thus, for example, **enlist** and **listen** are conjugates. A word is said to be *Lyndon* if it is

primitive and lexicographically least among all its conjugates. Thus, for example, **academy** is Lyndon, while **googol** and **googoo** are not. A classical theorem is that a finite word is Lyndon if and only if it is lexicographically less than each of its proper suffixes [7].

We now turn to (right-) infinite words. We write an infinite word in boldface, as $\mathbf{x} = a_0a_1a_2\cdots$ and use indexing starting at 0. For $i \leq j + 1$, we let $[i..j]$ denote the set $\{i, i + 1, \dots, j\}$. (If $i = j + 1$ we get the empty set.) We let $\mathbf{x}[i..j]$ denote the word $a_i a_{i+1} \cdots a_j$. Similarly, $[i..\infty]$ denotes the infinite set $\{i, i + 1, \dots\}$ and $\mathbf{x}[i..\infty]$ denotes the infinite word $a_i a_{i+1} \cdots$.

An infinite word or sequence $\mathbf{x} = a_0a_1a_2\cdots$ is said to be *k-automatic* if there is a deterministic finite automaton (with outputs associated with the states) that, on input n expressed in base k , reaches a state q with output $\tau(q)$ equal to a_n . For more details, see [5] or [3]. In several previous papers [1, 4, 14, 16, 8], we have developed a technique to show that many properties of automatic sequences are decidable. The fundamental tool is the following:

Theorem 1. *Let $P(n)$ be a predicate associated with a k -automatic sequence \mathbf{x} , expressible using addition, subtraction, comparisons, logical operations, indexing into \mathbf{x} , and existential and universal quantifiers. Then there is a computable finite automaton accepting the base- k representations of those n for which $P(n)$ holds. Furthermore, we can decide if $P(n)$ holds for at least one n , or for all n , or for infinitely many n .*

If a predicate is constructed as in the previous theorem, we just say it is “expressible”. Any expressible predicate is decidable. As an example, we prove

Theorem 2. *Let \mathbf{x} be a k -automatic sequence. The predicate $P(i, j)$ defined by “ $\mathbf{x}[i..j]$ is primitive” is expressible.*

Proof. (due to Luke Schaeffer) It is easy to see that a word is a power if and only if it is equal to some cyclic shift of itself, other than the trivial shift. Thus a word is a power if and only if there is a d , $0 < d < j - i + 1$, such that $\mathbf{x}[i..j-d] = \mathbf{x}[i+d..j]$ and $\mathbf{x}[j-d+1..j] = \mathbf{x}[i..i+d-1]$. A word is primitive if there is no such d . \square

Theorem 3. *Let \mathbf{x} be a k -automatic sequence. The predicate $LL(i, j, m, n)$ defined by “ $\mathbf{x}[i..j] < \mathbf{x}[m..n]$ ” is expressible.*

Proof. We have $\mathbf{x}[i..j] < \mathbf{x}[m..n]$ if and only if either

- (a) $j - i < n - m$ and $\mathbf{x}[i..j] = \mathbf{x}[m..m + j - i]$; or
- (b) there exists $t < \min(j - i, n - m)$ such that $\mathbf{x}[i..i + t] = \mathbf{x}[m..m + t]$ and $\mathbf{x}[i + t + 1] < \mathbf{x}[m + t + 1]$.

\square

Theorem 4. *Let \mathbf{x} be a k -automatic sequence. The predicate $L(i, j)$ defined by “ $\mathbf{x}[i..j]$ is a Lyndon word” is expressible.*

Proof. It suffices to check that $\mathbf{x}[i..j]$ is lexicographically less than each of its proper suffixes, that is, that $LL(i, j, i', j)$ holds for all i' with $i < i' \leq j$. \square

We can extend the definition of lexicographic order to infinite words in the obvious way. We can extend the definition of Lyndon words to (right-) infinite words as follows: an infinite word $\mathbf{x} = a_0a_1a_2\cdots$ is Lyndon if it is lexicographically less than all its suffixes $\mathbf{x}[j..\infty] = a_ja_{j+1}\cdots$ for $j \geq 1$. Then we have the following theorems.

Theorem 5. *Let \mathbf{x} be a k -automatic sequence. The predicate $LL_\infty(i, j)$ defined by “ $\mathbf{x}[i..\infty] < \mathbf{x}[j..\infty]$ ” is expressible.*

Proof. This is equivalent to $\exists t \geq 0$ such that $\mathbf{x}[i..i+t-1] = \mathbf{x}[j..j+t-1]$ and $\mathbf{x}[i+t] < \mathbf{x}[j+t]$. \square

Theorem 6. *Let \mathbf{x} be a k -automatic sequence. The predicate $L_\infty(i)$ defined by “ $\mathbf{x}[i..\infty]$ is an infinite Lyndon word” is expressible.*

Proof. This is equivalent to $LL_\infty(i, j)$ holding for all $j > i$. \square

2 Lyndon factorization

Siromoney et al. [12] proved that every infinite word $\mathbf{x} = a_0a_1a_2\cdots$ can be factorized uniquely in exactly one of the following two ways:

- (a) as $\mathbf{x} = w_1w_2w_3\cdots$ where each w_i is a finite Lyndon word and $w_1 \geq w_2 \geq w_3\cdots$; or
- (b) as $\mathbf{x} = w_1w_2w_3\cdots w_r\mathbf{w}$ where w_i is a finite Lyndon word for $1 \leq i \leq r$, and \mathbf{w} is an infinite Lyndon word, and $w_1 \geq w_2 \geq \cdots \geq w_r \geq \mathbf{w}$.

If (a) holds we say that the Lyndon factorization of \mathbf{x} is infinite; otherwise we say it is finite.

Ido and Melançon [11, 10] gave an explicit description of the Lyndon factorization of the Thue-Morse word \mathbf{t} and the period-doubling sequence (among other things). (Recall that the Thue-Morse word is given by $\mathbf{t}[n] =$ the number of 1’s in the binary expansion of n , taken modulo 2.) For the Thue-Morse word, this factorization is given by

$$\mathbf{t} = w_1w_2w_3w_4\cdots = (011)(01)(0011)(00101101)\cdots,$$

where each term in the factorization, after the first, is double the length of the previous. Séébold [15] and Černý generalized these results to other related automatic sequences.

In this section, generalizing the work of Ido, Melançon, Séébold, and Černý, we prove that the Lyndon factorization of a k -automatic sequence is itself k -automatic. Of course, we need to explain how the factorization is encoded. The easiest and most natural way to do this is to use an infinite word over $\{0, 1\}$, where the 1’s indicate the positions where a new

term in the factorization begins. Thus the i 'th 1, for $i \geq 0$, appears at index $|w_1 w_2 \cdots w_i|$. For example, for the Thue-Morse word, this encoding is given by

$$100101000100000001 \cdots.$$

If the factorization is infinite, then there are infinitely many 1's in its encoding; otherwise there are finitely many 1's.

In order to prove the theorem, we need a number of results. We draw a distinction between a *factor* f of \mathbf{x} (which is just a word) and an *occurrence* of that factor (which specifies the exact position at which f occurs). For example, in the Thue-Morse word \mathbf{t} , the factor 0110 occurs as $\mathbf{x}[0..3]$ and $\mathbf{x}[11..15]$ and many other places. We call $[0..3]$ and $[11..15]$, and so forth, the *occurrences* of 0110. An occurrence is said to be Lyndon if the word at that position is Lyndon. We say an occurrence $O_1 = [i..j]$ is *inside* an occurrence $O_2 = [i'..j']$ if $i' \leq i$ and $j' \geq j$. If, in addition, either $i' < i$ or $j < j'$ (or both), then we say O_1 is *strictly inside* O_2 . These definitions are easily extended to the case where j or j' are equal to ∞ , and they correspond to the predicates I (inside) and SI (strictly inside) given below:

$$\begin{aligned} I(i, j, i', j') & \text{ is } i' \leq i \text{ and } j' \geq j \\ SI(i, j, i', j') & \text{ is } I(i, j, i', j') \text{ and } ((i' < i) \text{ or } (j < j')) \end{aligned}$$

An infinite Lyndon factorization

$$\mathbf{x} = w_1 w_2 w_3 \cdots$$

then corresponds to an infinite sequence of occurrences

$$[i_1..j_1], [i_2..j_2], \cdots$$

where $w_n = \mathbf{x}[i_n..j_n]$ and $i_{n+1} = j_n + 1$ for $n \geq 1$, while a finite Lyndon factorization

$$\mathbf{x} = w_1 w_2 \cdots w_r \mathbf{w}$$

corresponds to a finite sequence of occurrences

$$[i_1..j_1], [i_2..j_2], \dots, [i_r..j_r], [i_{r+1}..\infty]$$

where $w_n = \mathbf{x}[i_n..j_n]$ and $i_{n+1} = j_n + 1$ for $1 \leq n \leq r$.

Theorem 7. *Let \mathbf{x} be an infinite word. Every Lyndon occurrence in \mathbf{x} appears inside a term of the Lyndon factorization of \mathbf{x} .*

Proof. We prove the result for infinite Lyndon factorizations; the result for finite factorizations is exactly analogous.

Suppose the factorization is $\mathbf{x} = w_1 w_2 w_3 \cdots$. It suffices to show that no Lyndon occurrence can span the boundary between two terms of the factorization. Suppose, contrary to what we want to prove, that $uw_i w_{i+1} \cdots w_j v$ is a Lyndon word for some u that is a nonempty

suffix of w_{i-1} (possibly equal to w_{i-1}), and v that is a nonempty prefix of w_{j+1} (possibly equal to w_{j+1}), and $i \leq j + 1$. (If $i = j + 1$ then there are no w_i 's at all between u and v .)

Since u is a suffix of w_{i-1} and w_{i-1} is Lyndon, we have $u \geq w_{i-1}$. On the other hand, by the Lyndon factorization definition we have $w_{i-1} \geq w_i \geq \dots \geq w_j \geq w_{j+1}$. But v is a prefix of w_{j+1} , so just by the definition of lexicographic ordering we have $w_{j+1} \geq v$. Putting this all together we get $u \geq v$. So $ux \geq v$ for all words x .

On the other hand, since $uw_i \dots w_j v$ is Lyndon, it must be lexicographically less than any proper suffix — for instance, v . So $uw_i \dots w_j v < v$. Take $x = w_i \dots w_j v$ to get a contradiction with the conclusion in the previous paragraph. \square

Corollary 8. *The occurrence $[i..j]$ corresponds to a term in the Lyndon factorization of \mathbf{x} if and only if*

- (a) $[i..j]$ is Lyndon; and
- (b) $[i..j]$ does not occur strictly inside any other Lyndon occurrence.

Proof. Suppose $[i..j]$ corresponds to a term w_n in the Lyndon factorization of \mathbf{x} . Then evidently $[i..j]$ is Lyndon. If it occurred strictly inside some other Lyndon occurrence, say $[i'..j']$, then we know from Theorem 7 that $[i'..j']$ itself lies inside some w_m , so $[i..j]$ must lie strictly inside w_m , which is clearly impossible.

Now suppose $[i..j]$ is Lyndon and does not occur strictly inside any other Lyndon occurrence. From Theorem 7 $[i..j]$ must occur inside some term of the factorization $[i'..j']$. If $[i..j] \neq [i'..j']$ then $[i..j]$ lies strictly inside $[i'..j']$, a contradiction. So $[i..j] = [i'..j']$ and hence corresponds to a term of the factorization. \square

Corollary 9. *The predicate $LF(i, j)$ defined by “ $[i..j]$ corresponds to a term of the Lyndon factorization of \mathbf{x} ” is expressible.*

Proof. Indeed, by Corollary 8, the predicate $LF(i, j)$ can be defined by

$$L(i, j) \text{ and } \forall i', j' (SI(i, j, i', j') \implies \neg L(i', j')).$$

\square

We can now prove the main result of this section.

Theorem 10. *Using the encoding mentioned above, the Lyndon factorization of a k -automatic sequence is itself k -automatic.*

Proof. Using the technique of [1], we can create an automaton that on input i expressed in base k , guesses j and checks if $LF(i, j)$ holds. If so, it outputs 1 and otherwise 0. To get the last i in the case that the Lyndon factorization is finite, we also accept i if $L_\infty(i)$ holds. \square

We also have

Theorem 11. *Let \mathbf{x} be a k -automatic sequence. It is decidable if the Lyndon factorization of \mathbf{x} is finite or infinite.*

Proof. The construction given above in the proof of Theorem 10 produces an automaton that accepts finitely many distinct i (expressed in base k) if and only if the Lyndon factorization of \mathbf{x} is finite. \square

We programmed up our method and found the Lyndon factorization of the Thue-Morse sequence \mathbf{t} , the period-doubling sequence \mathbf{d} , the paperfolding sequence \mathbf{p} , and the Rudin-Shapiro sequence \mathbf{r} , and their negations. (The results for Thue-Morse and the period-doubling sequence were already given in [10], albeit in a different form.) Recall that the period-doubling sequence is defined by $\mathbf{p}[n] = |\mathbf{t}[n+1] - \mathbf{t}[n]|$. The paperfolding sequence $\mathbf{p} = 0010011 \dots$ arises from the limit of the sequence (f_n) , where $f_0 = 0$ and $f_{n+1} = f_n 0 \overline{f_n}^R$, where R denotes reversal and \bar{x} maps 0 to 1 and 1 to 0. Finally, the Rudin-Shapiro sequence \mathbf{r} is defined by $\mathbf{r}[n] =$ the number of (possibly overlapping) occurrences of 11 in the binary expansion of n , taken modulo 2. The results are given in the theorem below.

Theorem 12. *The occurrences corresponding to the Lyndon factorization of each word is as follows:*

- the Thue-Morse sequence \mathbf{t} : $[0..2], [3..4], [5..8], [9..16], [17..32], \dots, [2^i + 1..2^{i+1}], \dots$;
- the negated Thue-Morse sequence $\bar{\mathbf{t}}$: $[0..0], [1..\infty]$;
- the Rudin-Shapiro sequence \mathbf{r} : $[0..6], [7..14], [15..30], \dots, [2^i - 1..2^{i+1} - 2], \dots$;
- the negated Rudin-Shapiro sequence $\bar{\mathbf{r}}$: $[0..0], [1..1], [2..2], [3..10], [11..42], [43..46], [47..174], \dots, [4^i - 4^{i-1} - 4^{i-2} - 1..4^i - 4^{i-1} - 2], [4^i - 4^{i-1} - 1..4^{i+1} - 4^i - 4^{i-1} - 1], \dots$;
- the paperfolding sequence \mathbf{p} : $[0..6], [7..14], [15..30], \dots, [2^i - 1..2^{i+1} - 2], \dots$;
- the negated paperfolding sequence $\bar{\mathbf{p}}$: $[0..0], [1..1], [2..4], [5..9], [10..20], [21..84], [85..340], \dots, [(4^i - 1)/3..4(4^i - 1)/3], \dots$;
- the period-doubling sequence \mathbf{d} : $[0..0], [1..4], [5..20], [21..84], \dots, [(4^i - 1)/3..4(4^i - 1)/3], \dots$;
- the negated period-doubling sequence $\bar{\mathbf{d}}$: $[0..1], [2..9], [10..41], [42..169], \dots, [2(4^i - 1)/3..2(4^{i+1} - 1)/3 - 1], \dots$

3 Enumeration

There is a useful generalization of k -automatic sequences to sequences over \mathbb{N} , the non-negative integers. A sequence $(a_n)_{n \geq 0}$ over \mathbb{N} is called k -regular if there exist vectors u and v and a matrix-valued morphism μ such that $a_n = u\mu(w)v$, where w is the base- k representation of n . For more details, see [2].

The subword complexity function $\rho(n)$ of an infinite sequence \mathbf{x} counts the number of distinct length- n factors of \mathbf{x} . There are also many variations, such as counting the number of palindromic factors or unbordered factors. If \mathbf{x} is k -automatic, then all three of these are k -regular sequences [1]. We now show that the same result holds for primitive or Lyndon factors.

Theorem 13. *The function counting the number of length- n primitive (resp., Lyndon) factors of a k -automatic sequence \mathbf{x} is k -regular.*

Proof. By the results of [4], it suffices to show that there is an automaton accepting the base- k representations of pairs (n, i) such that the number of i 's associated with each n equals the number of primitive (resp., Lyndon) factors of length n .

To do so, it suffices to show that the predicate $P(n, i)$ defined by “the factor of length n beginning at position i is primitive (resp., Lyndon) and is the first occurrence of that factor in \mathbf{x} ” is expressible. This is just

$$P(i, i + n - 1) \quad \text{and} \quad \forall j < i \quad \mathbf{x}[i..i + n - 1] \neq \mathbf{x}[j..j + n - 1],$$

(resp.,

$$L(i, i + n - 1) \quad \text{and} \quad \forall j < i \quad \mathbf{x}[i..i + n - 1] \neq \mathbf{x}[j..j + n - 1]).$$

□

We used our method to compute these sequences for the Thue-Morse sequence, and the results are given below.

Theorem 14. *Let $\rho_{\mathbf{t}}^L(n)$ denote the number of Lyndon factors of length n of the Thue-Morse sequence. Then*

$$\rho_{\mathbf{t}}^L(n) = \begin{cases} 1, & \text{if } n = 2^k \text{ or } 5 \cdot 2^k \text{ for } k \geq 1; \\ 2, & \text{if } n = 1 \text{ or } n = 5 \text{ or } n = 3 \cdot 2^k \text{ for } k \geq 0; \\ 0, & \text{otherwise.} \end{cases}$$

Theorem 15. *Let $\rho_{\mathbf{t}}^P(n)$ denote the number of primitive factors of length n of the Thue-Morse sequence. Then*

$$\rho_{\mathbf{t}}^P(n) = \begin{cases} 3 \cdot 2^t - 4, & \text{if } n = 2^t; \\ 4n - 2^t - 4, & \text{if } 2^t + 1 \leq n < 3 \cdot 2^{t-1}; \\ 5 \cdot 2^t - 6, & \text{if } n = 3 \cdot 2^{t-1}; \\ 2n + 2^{t+1} - 2, & \text{if } 3 \cdot 2^{t-1} < n < 2^{t+1}. \end{cases}$$

We can also state a similar result for the Rudin-Shapiro sequence.

Theorem 16. *Let $\rho_{\mathbf{r}}^L(n)$ denote the number of Lyndon factors of length n of the Rudin-Shapiro sequence. Then $\rho_{\mathbf{r}}^L(n) \leq 8$ for all n . This sequence is 2-automatic and there is an automaton of 2444 states that generates it.*

Proof. The proof was carried out by machine computation, and we briefly summarize how it was done.

First, we created an automaton A to accept all pairs of integers (n, i) , represented in base 2, such that the factor of length n in \mathbf{r} , starting at position i , is a Lyndon factor, and is the first occurrence of that factor in \mathbf{r} . Thus, the number of distinct integers i associated with each n is $\rho_{\mathbf{r}}^L(n)$. The automaton A has 102 states.

Using the techniques in [4], we then used A to create matrices M_0 and M_1 of dimension 102×102 , and vectors v, w such that $vM_x w = \rho_{\mathbf{r}}^L(n)$, if x is the base-2 representation of n . Here if $x = a_1 a_2 \cdots a_i$, then by M_x we mean the product $M_{a_1} M_{a_2} \cdots M_{a_i}$.

From this we then created a new automaton A' where the states are products of the form vM_x for binary strings x and the transitions are on 0 and 1. This automaton was built using a breadth-first approach, using a queue to hold states whose targets on 0 and 1 are not yet known. Of course, this was not guaranteed to terminate, because *a priori* we did not know that $\rho_{\mathbf{r}}^L(n)$ is bounded. But the procedure did terminate at 2444 states, and the product of the vM_x corresponding to each state with w gives an integer less than or equal to 8, thus proving the desired result and also providing an automaton to compute $\rho_{\mathbf{r}}^L(n)$. \square

4 Finite factorizations

Of course, the original Lyndon factorization was for finite words: every finite nonempty word x can be factored uniquely as a nonincreasing product $w_1 w_2 \cdots w_m$ of Lyndon words. We can apply this theorem to all prefixes of a k -automatic sequence. It is then natural to wonder if a *single* automaton can encode *all* the Lyndon factorizations of *all* finite prefixes. The answer is yes, as the following result shows.

Theorem 17. *Suppose \mathbf{x} is a k -automatic sequence. Then there is an automaton A accepting*

$$\{(n, i)_k : \text{the Lyndon factorization of } \mathbf{x}[0..n-1] \text{ is } w_1 w_2 \cdots w_m \text{ with } w_m = \mathbf{x}[i..n-1]\}.$$

Proof. As is well-known [7], if $w_1 w_2 \cdots w_m$ is the Lyndon factorization of x , then w_m is the lexicographically least suffix of x . So to accept $(n, i)_k$ we find t such that $\mathbf{x}[i..n-1] < \mathbf{x}[j..n-1]$ for $0 \leq j < n$ and $i \neq j$. \square

Given A , we can find the complete factorization of any prefix $\mathbf{x}[0..n-1]$ by using this automaton to find the appropriate l (as described in [9]) and then replacing n with l .

We carried out this construction for the Thue-Morse sequence, and the result is shown below in Figure 4.

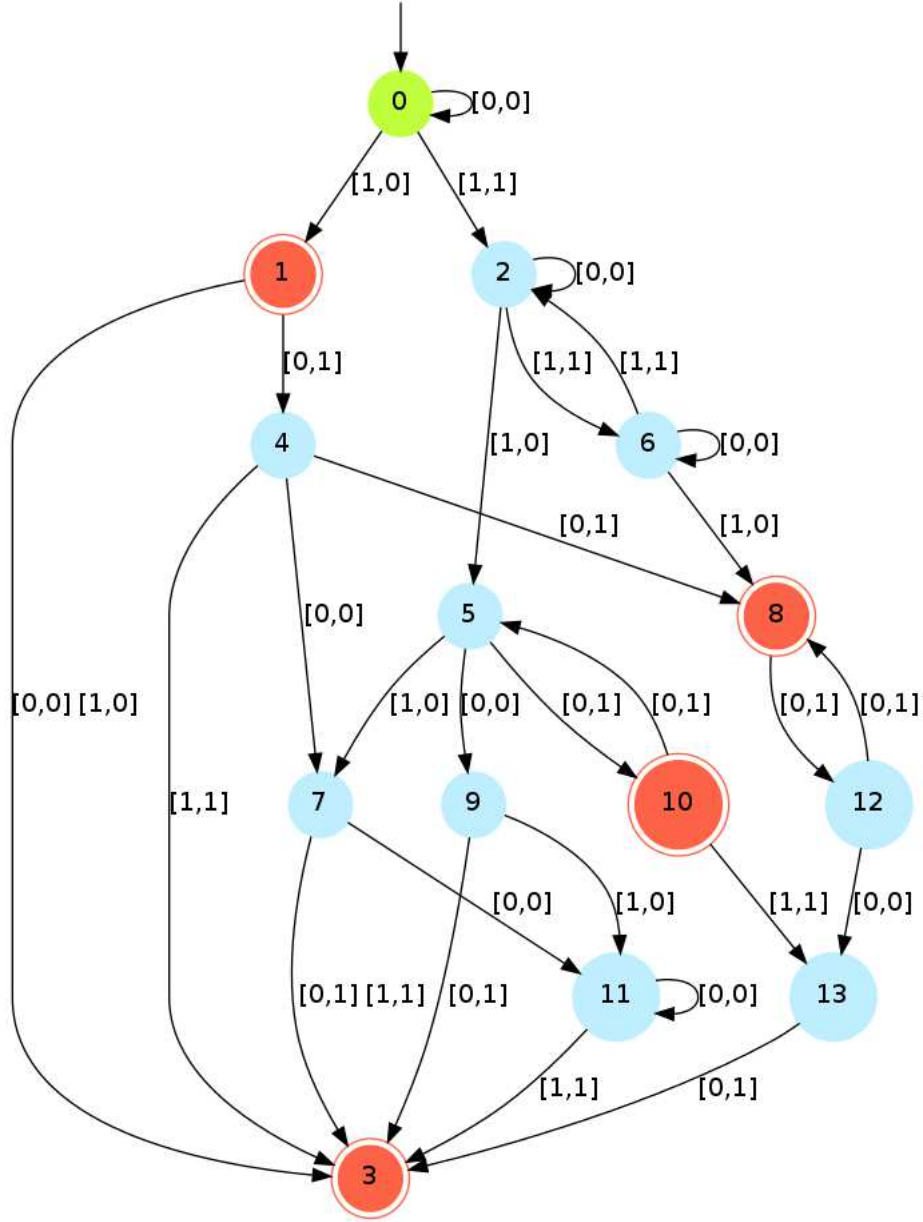


Figure 1: A finite automaton accepting the base-2 representation of (n, i) such that the Lyndon factorization of $\mathbf{t}[0..n-1]$ ends in the term $\mathbf{t}[i..n-1]$

In a similar manner, there is an automaton that encodes the factorization of *every* factor of a k -automatic sequence:

Theorem 18. *Suppose \mathbf{x} is a k -automatic sequence. Then there is an automaton A' accept-*

ing

$\{(i, j, l)_k : \text{the Lyndon factorization of } \mathbf{x}[i..j-1] \text{ is } w_1 w_2 \cdots w_m \text{ with } w_m = \mathbf{x}[l..n-1]\}.$

We calculated A' for the Thue-Morse sequence using our method. It is a 34-state machine and is displayed in Figure 4.

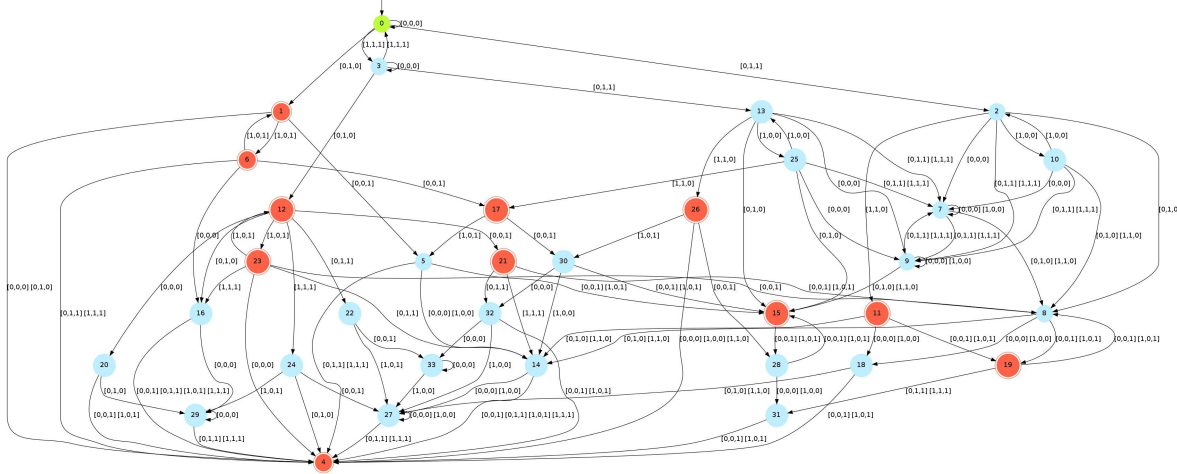


Figure 2: A finite automaton accepting the base-2 representation of (i, j, l) such that the Lyndon factorization of $\mathbf{t}[i..j-1]$ ends in the term $\mathbf{t}[l..j-1]$

Another quantity of interest is the number of terms in the Lyndon factorization of each prefix.

Theorem 19. *Let x be a k -automatic sequence. Then the sequence $(f(n))_{n \geq 0}$ defined by*

$$f(n) = \text{the number of terms in the Lyndon factorization of } \mathbf{x}[0..n]$$

is k -regular.

Proof. We construct an automaton to accept

$$\{(n, i) : \exists j \leq n \text{ such that } L(i, j) \text{ and if } SI(i, j, i', j') \text{ and } 0 \leq i' \leq j' \leq n \text{ then } \neg L(i', j')\}.$$

□

For the Thue-Morse sequence the corresponding sequence satisfies the relations

$$\begin{aligned}
f(4n+1) &= -f(2n) + f(2n+1) + f(4n) \\
f(8n+2) &= -f(2n) + f(4n) + f(4n+2) \\
f(8n+3) &= -f(2n) + f(4n) + f(4n+3) \\
f(8n+6) &= -f(2n) - f(4n+2) + 3f(4n+3) \\
f(8n+7) &= -f(2n) + 2f(4n+3) \\
f(16n) &= -f(2n) + f(4n) + f(8n) \\
f(16n+4) &= -f(2n) + f(4n) + f(8n+4) \\
f(16n+8) &= -f(2n) + f(4n+3) + f(8n+4) \\
f(16n+12) &= -f(2n) - 2f(4n+2) + 3f(4n+3) + f(8n+4)
\end{aligned}$$

for $n \geq 1$, which allows efficient calculation of this quantity.

References

- [1] J.-P. Allouche, N. Rampersad, and J. Shallit. Periodicity, repetitions, and orbits of an automatic sequence. *Theor. Comput. Sci.* **410** (2009), 2795–2803.
- [2] J.-P. Allouche and J. O. Shallit. The ring of k -regular sequences, *Theor. Comput. Sci.* **98** (1992), 163–197.
- [3] J.-P. Allouche and J. Shallit. *Automatic Sequences: Theory, Applications, Generalizations*, Cambridge, 2003.
- [4] E. Charlier, N. Rampersad, and J. Shallit. Enumeration and decidable properties of automatic sequences. In G. Mauri and A. Leporati, eds., *Developments in Language Theory — 15th International Conference, DLT 2011*, Vol. 6795, pp. 165–179, Springer-Verlag, 2011.
- [5] A. Cobham. Uniform tag sequences. *Math. Systems Theory* **6** (1972), 164–192.
- [6] A. Černý. Lyndon factorization of generalized words of Thue, *Disc. Math. Theor. Comput. Sci.* **5** (2002), 17–46.
- [7] J. P. Duval. Factorizing words over an ordered alphabet. *J. Algorithms* **4** (1983), 363–381.
- [8] D. Goč, D. Henshall, and J. Shallit. Automatic theorem-proving in combinatorics on words. In N. Moreira and R. Reis, eds., *CIAA 2012*, Lect. Notes in Comput. Sci., Vol. 7381, Springer, 2012, pp. 180–191.
- [9] D. Goč, L. Schaeffer, and J. Shallit. The subword complexity of k -automatic sequences is k -synchronized. Preprint, June 23 2012. Available from <http://arxiv.org/abs/1206.5352>.

- [10] A. Ido and G. Melançon. Lyndon factorization of the Thue-Morse word and its relatives. *Discrete Math. Theoret. Comput. Sci.* **1** (1997), 43–52.
- [11] G. Melançon. Lyndon factorization of infinite words. In C. Puech and R. Reischuk, eds., *STACS 96, 13th Annual Symposium on Theoretical Aspects of Computer Science*, Lect. Notes in Comput. Sci., Vol. 1046, Springer, 1996, pp. 147–154.
- [12] R. Siromoney, L. Matthew, V. R. Dare, and K. G. Subramanian. Infinite Lyndon words. *Info. Process. Letters* **50** (1994), 101–104.
- [13] E. Rowland and J. Shallit. k -automatic sets of rational numbers. In A. H. Dediu and C. Martín-Vide, eds., *LATA 2012 Proceedings*, Lect. Notes in Comp. Sci., Vol. 7183, Springer, 2012, pp. 490–501.
- [14] L. Schaeffer and J. Shallit. The critical exponent is computable for automatic sequences. *Int. J. Found. Comput. Sci.*, to appear, 2012.
- [15] P. Séébold. Lyndon factorization of the Prouhet words. *Theor. Comput. Sci.* **307** (2003), 179–197.
- [16] J. Shallit. The critical exponent is computable for automatic sequences. In P. Ambroz, S. Holub, and Z. Másaková, eds., *Proceedings 8th International Conference Words 2011*, Vol. 63 of *Elect. Proc. Theor. Comput. Sci.*, pp. 231–239, 2011.